BIS Innovation Hub

Project Aurum

A Prototype for Two-tier Central Bank Digital Currency (CBDC)

CBDC Central Bank Software Design

October 2022

Bank for International Settlements (BIS)

Contents

Revision History

Version	Date	Description
1.0	28 Oct 2022	First release.

1 Software Overview

This document provides detail of the Central Bank server software for the CBDC system. Central Bank is in charge of overseeing the interbank system and the RTGS system. In the interbank system, it runs a blockchain peer node for the purpose of having direct access to the blockchain ledger containing interbank system transactions, and for the purpose of executing CBDC transactions. It manages the RTGS system and processes requests from banks for exchanges between CBDC and RTGS accounts.

Central Bank is also responsible for assigning validators to the banks for validating their eWallet system transactions involving stablecoin and CBDC-token.

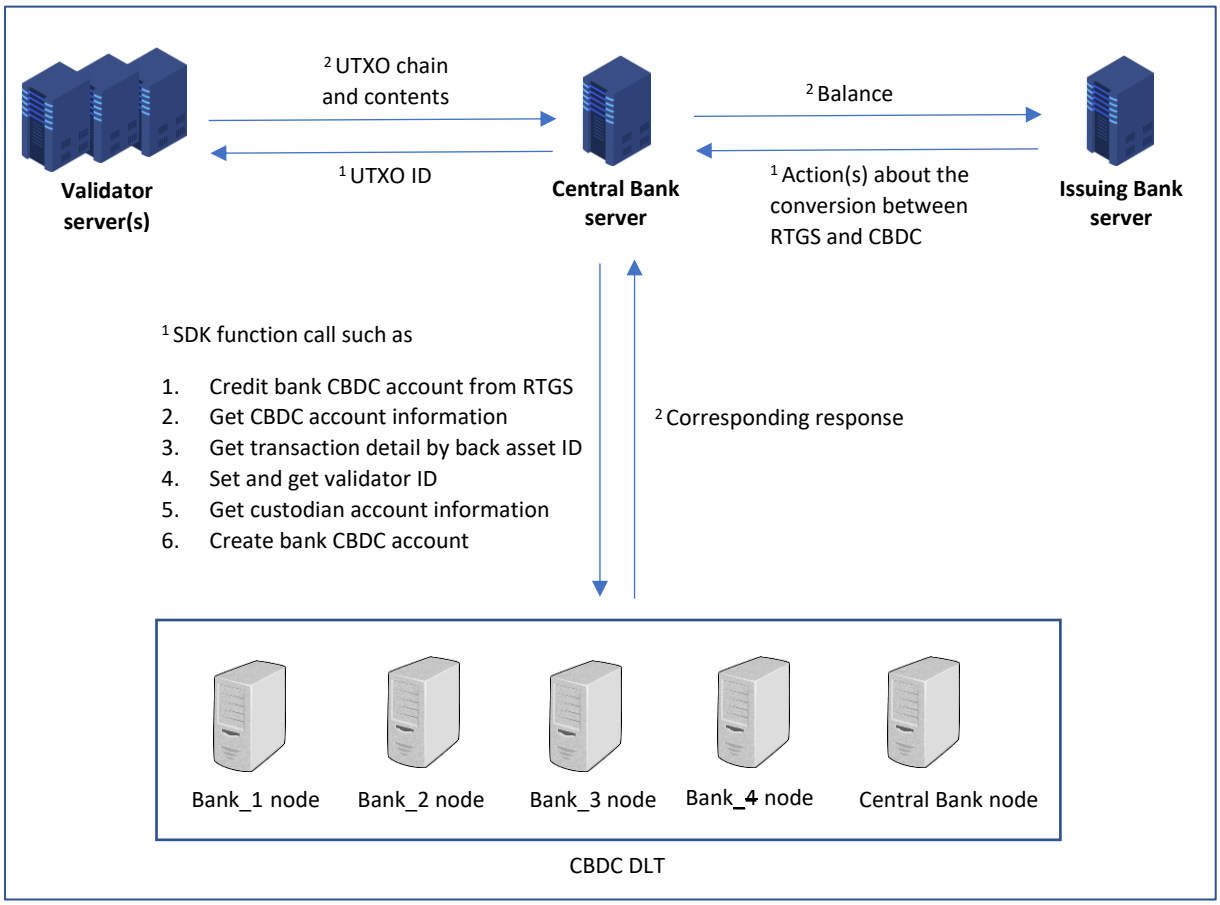


Figure 1: System Architecture

2 Software Block Diagram

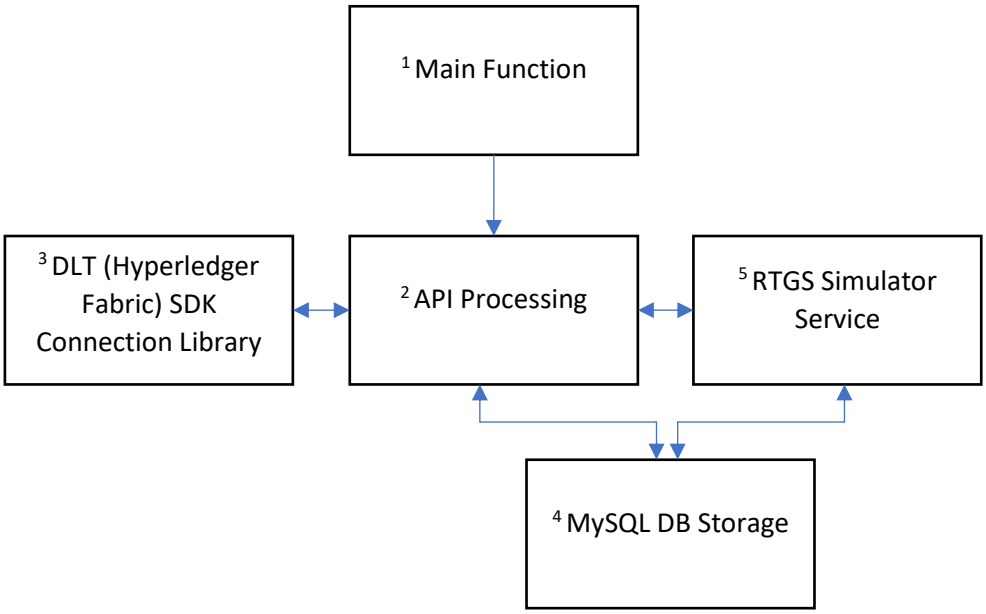


Figure 2: Software Block Diagram

After the [1] main function is started, APIs can be called. For some API calls, [3] DLT (Hyperledger Fabric) SDK Connection Library is needed to connect the DLT and do some operation like "set & get the validator ID". [4] MySQL DB Storage is designed to store the user login related work and the RTGS transaction records. [5] RTGS Simulator Service is a customizable function to do account operation between RTGS and CBDC.

3 Function Description

3.1 API Processing

Refer to the Central Bank API document - cbdc-central-bank-api.html

3.2 MySQL DB Storage

Relational database storage to store tables for:

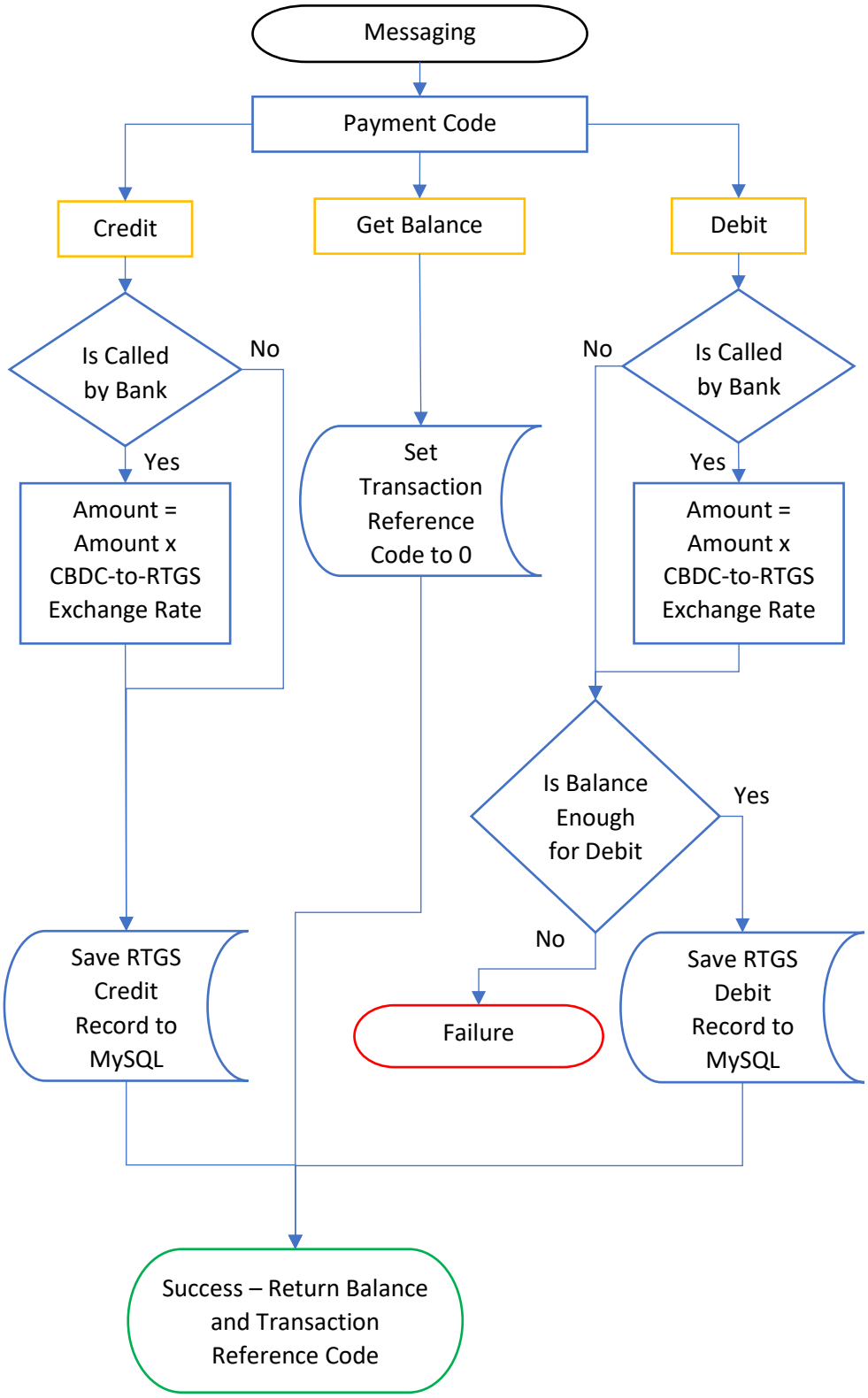
1. User management
2. Token management
3. RTGS records

3.3 RTGS Simulator Service

This part describes the messaging functionalities by flow chart.

3.3.1 Messaging

- Acting as a middleware to process the RTGS account operation such as the conversion between RTGS and CBDC


Messaging
Payment Code
Credit
Get Balance
Debit
Is Called by Bank
No
Yes
Amount = Amount x CBDC-to-RTGS Exchange Rate
Set Transaction Reference Code to 0
Is Called by Bank
No
Yes
Amount = Amount x CBDC-to-RTGS Exchange Rate
Is Balance Enough for Debit
Yes
No
Save RTGS Credit Record to MySQL
Failure
Save RTGS Debit Record to MySQL
Success – Return Balance and Transaction Reference Code

-

4 Customizable Functions

4.1 RTGS_SIMULATOR_SERVICE_MESSAGING

Functionality:

- Messaging function to be called by Central Bank to service bank requests for RTGS accounts operation

Class Name:

- RTGSSimulatorService

Function Name:

- message

Input Parameters:

- Payment Code: (Enum - Simulator Message Type)

  Values:
  DEBIT = Debit RTGS account (for credit to CBDC account)
  CREDIT = Credit RTGS account (for debit from CBDC account)
  GET_BALANCE = Get RTGS account balance

- Amount: (BigDecimal)

  This field is required only for “Payment Code” parameter is DEBIT or CREDIT.

- Bank ID: (String)

  The ID (Public Key) of the bank that owns the RTGS account.

- Message: (String)

  To be customized. Currently there is not any verification checking between message and signature within this function.

- Signature: (String)

  To be customized. Currently there is not any verification checking between message and signature within this function.

- Remarks: (String)

  This field is not required.

- Is Called from Bank Server: (Boolean)

If this value is **TRUE**, this function is called from APIs
/bank*/cb_bank_credit_rtgs_for_redeemed_cbdc* &
/bank*/cb_bank_debit_rtgs_credit_cbdc*
=> "Amount" means CBDC amount.

Otherwise, this function is called from API
/staff*/cb_staff_credit_rtgs_account*
=> "Amount" means RTGS amount.

Output Parameters:

- Response (JSONObject)
  e.g.
  {"rtgsBalance": 12300, "txnRefCode": 0}

Note:
RTGS simulator will maintain a static counter called "RTGS transaction reference code counter" - "txnRefCode". It is initialized to a unique starting value. Whenever its Debit/Credit RTGS account messaging function is called, the counter value will be returned to the caller as the transaction reference code. The counter will then be incremented. If Payment Code is equal to GET_BALANCE, a transaction reference code of 0 is always returned.

5 Configuration Properties in application.properties

5.1 Summary Of Configuration Properties

Name	Default
cors.enabled	false
server.addr	127.0.0.1
server.host.name	cb.cbdc
server.port	8086
jwt.secret	NlLezCw2MNI
account.default.password	centr@18ank
account.default.password.for.admin	centr@18ank
account.default.admin.login.id	admin
rest.template.connect.protocol	https
custom.user.event.log	true
login.security.jwt.expired.time.in.sec	86400
login.security.jwt.refresh.expired.time.in.sec	86400
bank.map.file.path	resources/bankMap.json
hyperledger.fabric.channel.name	cbdcchannel
hyperledger.fabric.contract.name	cbdc
hyperledger.fabric.username	staff
hyperledger.fabric.org	cb
hyperledger.fabric.org.id	CbMSP
hyperledger.fabric.port	7054
hyperledger.fabric.wallet.location	resources/wallet
fabric.cert.folder.location	resources/fabricCert
day.tolerance.for.jwt.token.scheduler	7

default.scheduler.cron.time	0 0 0 * * *
retry.maxAttempts	5
retry.delay (milliseconds)	1000
exchange.rate.from.cbdc.to.rtgs	1
rtgs.cbdc.test.controller.enabled	true

5.1.1 cors.enabled

Cross-origin resource sharing (CORS) is a mechanism that allows restricted resources including API on a web page to be requested from another domain. If the setting is true, all resources and APIs can be accessed from another domain.

5.1.2 server.addr

This property is self-address.

5.1.3 server.host.name

This property is self-host name.

5.1.4 server.port

This property is self-port number.

5.1.5 jwt.secret

Use this secret to generate JWT token that is access control of API calling for each user.

5.1.6 account.default.password

Default account password for other banks.

5.1.7 account.default.password.for.admin

Default admin password.

5.1.8 account.default.admin.login.id

Default admin login ID.

5.1.9 rest.template.connect.protocol

To configure restful api connect protocol such as http or https.

5.1.10 custom.user.event.log

To configure custom event logger messages on/off.

5.1.11 login.security.jwt.expired.time.in.sec

To configure JWT expired time length in seconds.

5.1.12 login.security.jwt.refresh.expired.time.in.sec

To configure refresh JWT expired time length in seconds.

5.1.13 bank.map.file.path

This file path shows bank map location. The bank map file store all involvers' data in JSON format including their URL and public key information etc…

5.1.14 hyperledger.fabric.channel.name

To define hyperledger channel name.

5.1.15 hyperledger.fabric.contract.name

To define hyperledger contract name.

5.1.16 hyperledger.fabric.username

To define hyperledger user name who will be representer for all transaction submissions.

5.1.17 hyperledger.fabric.org

To define which hyperledger organization will be connected.

5.1.18 hyperledger.fabric.org.id

To define hyperledger organization identity.

5.1.19 hyperledger.fabric.port

To define hyperledger organization port number.

5.1.20 hyperledger.fabric.wallet.location

To define hyperledger wallet folder location.

5.1.21 fabric.cert.folder.location

To define hyperledger certificate folder location.

5.1.22 day.tolerance.for.jwt.token.scheduler

To keep JWT in the system within the number of tolerance days. If JWT expired and stored exceed the tolerance days. System will erase them in scheduler which defined in another property (default.scheduler.cron.time).

5.1.23 default.scheduler.cron.time

Scheduling a cron job to check and delete expired JWT.

5.1.24 retry.maxAttempts

Retry max number of attempts

5.1.25 retry.delay

Retry time interval after fail.

5.1.26 exchange.rate.from.cbdc.to.rtgs

To configure the exchange rate from CBDC to RTGS.

5.1.27 rtgs.cbdc.test.controller.enabled

To enable abnormal cases test controller, for example, emulating a long delay in middle of RTGS and CBDC exchange.

6 Major Data Structures

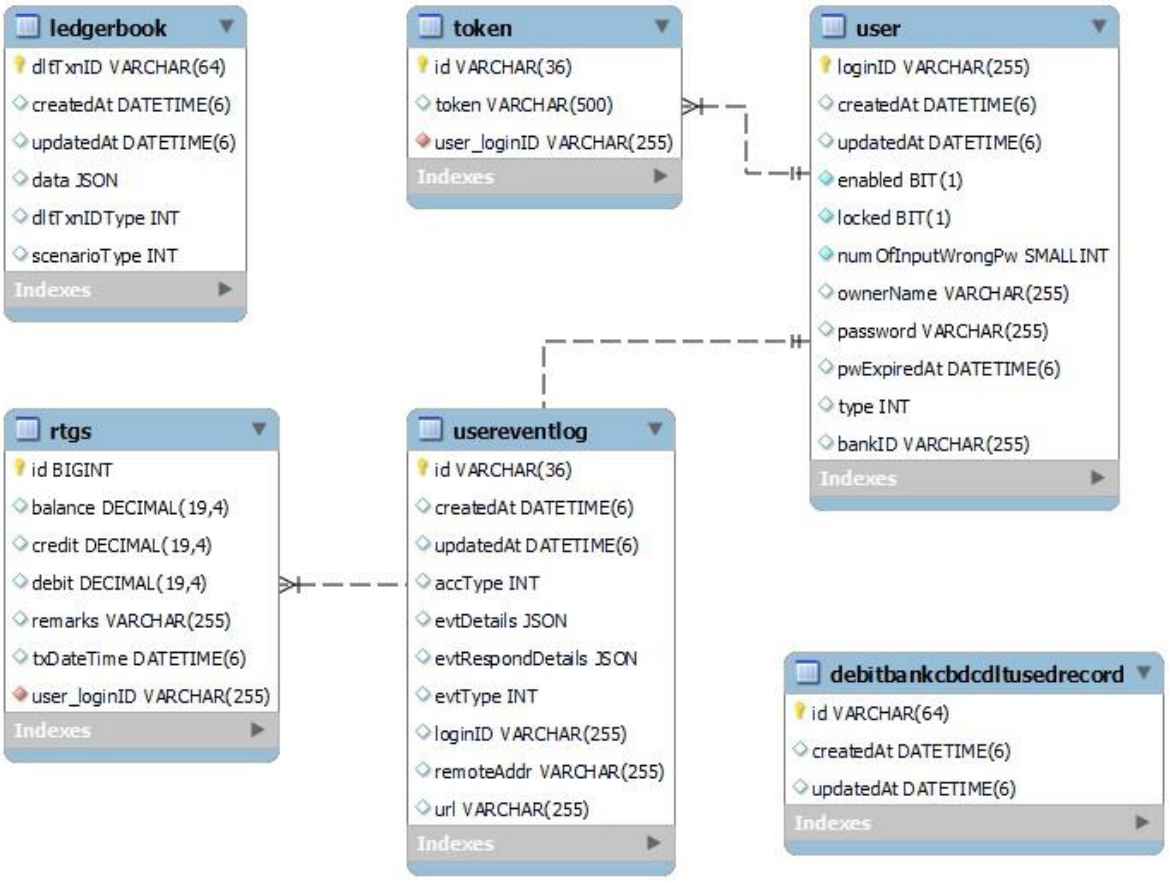


Figure 3: Central Bank Database Entity-Relationship Diagram

6.1 Entity Description

Entity Name	Description
debitbankcbdcdltusedrecord	Record DLT debit CBDC transaction ID that have been used by banks to request for RTGS.
ledgerbook	Record all exception or timeout DLT transaction IDs.
rtgs	Record bank's rtgs history.
token	Save users' login token.
user	Save users' meta data.
usereventlog	Save users' activities event log, mainly save for their API calling history.

7 Source Code Archive Structure

Source code	Release Sub-Folder
Central Bank server	1) cbdc/components/database/keeperCentralBank Database library for controlling validator server self-database including MySQL 2) cbdc/centralBank Central Bank server 3) cbdc/components/utility Database library for Hyperledger Fabric SDK library